IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

Applicant(s):	Harm Sluiman		
Assignee:	International Business Machines Corporation		
Title:	Automation and Isolation of Software Component Testing		
Serial No.:	09/772,650	Filing Date:	January 30, 2001
Examiner:	Insun Kang	Group Art Unit:	2193
Docket No.:	CA920000042US1	Customer No.:	61136

Austin, Texas
June 22, 2006

MAIL STOP AF
COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, VA 22313-1450

PRE-APPEAL BRIEF REQUEST FOR REVIEW AND STATEMENT OF REASONS

Sir:

Applicant requests review of the Final Rejection in the above-identified application. No amendments are being filed with the request. This request is being filed with a Notice of Appeal. The following sets forth a succinct, concise, and focused set of arguments for which the review is being requested.

CLAIM STATUS

Claims 1 - 8 stand rejected under 35 U.S.C. 103(a) as being unpatentable over Kobayashi, U.S. Patent No. 6,633,888 (Kobayashi) in view of Applicant's Admitted Prior Art (hereinafter APA).

REMARKS

The present invention, as set forth by claim 1, relates to a method for testing software test components. The method includes ascertaining a public interface of the software test component and creating a wrapper component for the software test component. Creating the wrapper component further includes defining a wrapper component interface to mirror the public interface of the test component and to receive calls to the software test component, defining the

wrapper component to delegate to the software test component by including calls to the public interface of the software test component within the wrapper component, inserting test code within the wrapper component to permit capture and playback of user interaction with the public interface of the software test component, and enabling a test case to use the wrapper component interface to pass the received calls to the software test component and to generate test data from the test code in the wrapper component.

Accordingly, Applicant's claims call for the wrapper to have an interface that mirrors the interface of the software test component. When a test case is applied to the wrapper, the wrapper interface is the same as would be the case for the software test component itself. The wrapper provides calls to the corresponding interface of the software test component and collects resulting data; but, it not a converter of signals for inconsistent interfaces.

Kobayashi generally relates to a method for creating and testing object oriented components with visual programming systems. More specifically, in Figure 2 of Kobayashi, the interface for test data applied to the proxy beans at box 216 is not the same as the interface of the universal transport API 206. For example, proxy beans have parameters represented by properties. It appears that box 206 translates the abnormal proxy bean data to the format recognized by normal class code at box 202, beans or that which is applied to the actual code at box 208. At col. 8, lines 23-27, Kobayashi teaches "each composite component in the application 216 can be tested within the visual builder by means of the universal transport API 206 which allows the code which implements the underlying objects and components 202 to be exercised under control of the proxy components." This does not disclose or suggest a test case interface mirrored on both sides of a wrapper. Kobayashi teaches adapters to allow a visual editor to support editing of proxy beans and then testing of the edited code by control exercised through the universal transport API 206. The interface changes from box to box in Figure 2. This deficiency isn't overcome by the other prior art.

When responding to Applicant's arguments, the examiner has set forth:

> [T]he examiner points out again that a wrapper in the Java programming language is an object that encapsulates and delegates to another object for altering its behavior or interface. According to the application (specification, page 10), this wrapper operates as a "proxy" for the actual component and the "delegation code is generated for the wrapper's proxy classes to pass calls through to the component being tested." The generation of this wrapper through reflection to mirror the test component such as the

limitations in the instant claim is possible through Java language features in the Java Bean specification. The instant specification also states that defining the wrapper component is possible by using tools such as the "introspection group of interfaces in the Java Bean specification." Such tools permit a wrapper generator to ascertain the members in the actual component to be used to define the proxy wrapper (page 10). Therefore, it is evident that the present invention simply uses the existing Java language features in the Java Bean specification to create a wrapper component, which acts as a proxy. That being said, Kobayashi's proxy bean acts as a delegate to the API of the actual bean and Kobayashi uses the parsing/extracting mechanism to determine/describe (i.e. introspections) and obtain (i.e. reflection) information about the members of a class such as the properties, methods, and constructs (i.e., "the parser/extractor 304 parses each constructor and each method and extracts any related fields, comments and parameter names," col. 8, lines 46-58). This extracting mechanism extends the conventional extraction process of "reflection" in the Java Bean specification so that the mechanism does not only determine the method parameters but also creates the "parameters to be converted to properties of the method bean created from the original method (col. 9 lines 1-19)." Using this extraction process (i.e. reflection), the APIs for all the classes can be retrieved and a proxy bean can be generated. The proxy component is to mirror the test component such as the wrapper in the instant claims. Therefore, Kobayashi discloses the limitation, defining a wrapper component interface to mirror the public interface of the software test component." Therefore, in view of the broadest reasonable interpretation, the rejection of the claims are considered proper and maintained. (Final office action dated March 23, 2006, pages 6, 7.)

Merely because the limitations of the claims is "possible" through features in the Java Bean specification does not mean that this same specification discloses or suggests such limitations. As discussed in the previous response, Applicant has found a clever way to tap into the data action during testing. The mirror interface allows the wrapper to support calls to access the interface of the software test component. Data is not changed to overcome an incompatibility, it is collected for evaluation. These features are not disclosed or suggested by Kobayashi or the APA.

More specifically, Kobayashi and the APA do not disclose or suggest a method for testing software test component where the method includes creating a wrapper component for the software test component where creating the wrapper component further includes *defining a wrapper component interface to mirror the public interface of the test component and to receive calls to the software test component*, defining the wrapper component to delegate to the software test component by *including calls to the public interface of the software test component within the wrapper component*, *inserting test code within the wrapper component to permit capture and playback of user interaction with the public interface of the software test component*, and *enabling a test case to use the wrapper component interface to pass the received calls to the*

software test component and to generate test data from the test code n the wrapper component, as required by claim 1. Claims 2 – 4 depend from claim 1 and are allowable for at least this reason. Claim 5 is a computer memory storing logic of similar scope to claim 1 and is allowable for similar reasons. Claims 6 – 8 depend from claim 5 and are allowable for at least this reason.

In view of the arguments set forth herein, the application is believed to be in condition for allowance and a notice to that effect is solicited. Nonetheless, should any issues remain that might be subject to resolution through a telephonic interview, please telephone the undersigned.

The Commissioner is authorized to deduct any additional fee which may be necessary or credit any overpayment to Deposit Account No. 090461.

I hereby certify that this correspondence is being submitted electronically to the COMMISSIONER FOR PATENTS on June 22, 2006.

/Stephen A. Terrile/

Attorney for Applicant(s)

Respectfully submitted,

/Stephen A. Terrile/

Stephen A. Terrile
Attorney for Applicant(s)
Reg. No. 32,946